Comp. & Maths. with Appls., Vol. 2, pp. 207–210. Pergamon Press 1976. Printed in Great Britain

AN ERROR ANALYSIS OF AN ALGORITHM FOR MARCUM'S Q-FUNCTION

PAUL J. NAHIN

Department of Electrical Engineering, University of New Hampshire, Durham, New Hampshire 03824, U.S.A.

Communicated by F. Y. Rodin

(*Received June* 1976)

Abstract—McGee's iterative algorithm for calculating Marcum's Q-Function is useful in many numerical studies in radar and communication systems. An analysis is presented that allows estimates on the computation time required, as a function of the desired accuracy, to support a call to a subroutine implementing this algorithm.

INTRODUCTION

Marcum's Q-function is defined by

$$Q(a, b) \triangleq \int_b^\infty v \exp\left[-\{v^2 + a^2\}/2\right] I_0(av)\, dv \tag{1}$$

where $I_0(\ )$ denotes the zero-order modified Bessel function[1]. Noticing that the integrand in eqn (1) is just the Rician probability density, we have the probabilistic interpretation of the Q-function as

$$Q(a, b) = \text{Prob}\,(v > b)$$

where v can, for example, represent the distance between a point in the plane to the origin when the cartesian co-ordinates are independent normal random variables of unit variance and expected values $a\cos\theta$ and $a\sin\theta$, θ an arbitrary angle[2]. It is therefore not surprising that the Q-function occurs in many important problems in communication theory and radar[3, 4].

Tables of this function are available and are useful for hand calculations[5]. However, it is often necessary to know $Q(a, b)$ to support calculations being performed automatically by a computer. Tables are not as convenient in this case, and it is desirable for $Q(a, b)$ to be calculated as needed, i.e. "on-line" to use an overworked colloquialism. As an extreme (but not particularly unusual) example, it might occur that $Q(a, b)$ is itself part of the integrand of an integral[6]. If this "super" integral is evaluated numerically by some technique, e.g. Simpson's Rule, then many calculations of $Q(a, b)$ may be necessary.

AN ITERATIVE ALGORITHM

To actually evaluate $Q(a, b)$ by quadrature methods can be very time consuming, particularly if several decimal places of accuracy are desired. To overcome this problem, iterative techniques have been developed that are fast and convenient to use[7, 8]. We examine here McGee's algorithm, a modified form of that of Brennan and Reed's. Both replace the zero-order Bessel function in eqn (1) by its power series expansion and carry out the integral term-by-term. From [8],

$$Q(a, b) = 1 - \sum_{k=0}^{\infty} f_k h_k \tag{2}$$

where

$$h_k = h_{k-1} + d_k$$

$$d_k = \frac{a^2}{2k} d_{k-1} \quad k \geq 1$$

$$f_k = \frac{b^2}{2(k+1)} f_{k-1}$$

$$d_0 = h_0 = \exp(-a^2/2)$$

$$f_0 = \frac{b^2}{2} \exp(-b^2/2)$$

McGee's algorithm is a "good" one because all the terms in the sum are positive, i.e. small errors will not cause instability in the method. It is clear that from the form of eqn (2), and as the sum is of all positive terms, that the McGee algorithm will always *over*estimate the Q-function. Given a and b, the expressions of eqn (2) are very easy to program. The question we examine in detail here is that of how many terms of the sum are required to compute $Q(a, b)$ to a predetermined accuracy.

ERROR ANALYSIS OF McGEE'S ALGORITHM

From the expressions for d_k in eqn (2) it is easy to show that

$$d_k = \left(\frac{a^2}{2}\right)^k \frac{1}{k!} \exp(-a^2/2), \quad k \geq 0$$

Combining this result with the expressions for h_k, it is again a simple matter to arrive at

$$h_k = \left[1 + \sum_{j=1}^{k} \left(\frac{a^2}{2}\right)^j \frac{1}{j!}\right] \exp(-a^2/2), \quad k \geq 0 \tag{3}$$

From eqn (3) it is obvious that h_k is monotonic increasing in k, i.e. $h_k < h_{k+1} \forall k$, and also that

$$\lim_{k \to \infty} h_k = 1 \forall a$$

This preliminary result means that the h_k will *not* play a significant role in determining how many terms must be retained in the sum of eqn (2) to achieve a given accuracy. Indeed, of the two arguments in $Q(a, b)$, only b is involved in our error analysis! For the sum in eqn (2) to converge, it must be true, of course, that

$$\lim_{k \to \infty} f_k = 0$$

Indeed, it is true that for k large enough, f_k does monotonically decrease toward zero. However, f_k exhibits a peaked behavior for "small" k, i.e. for some positive integer $\hat{k}$,

$$f_{\hat{k}-1} < f_{\hat{k}} \geq f_{\hat{k}+1}$$

To see this, the expressions for f_k in eqn (2) can be used to write

$$f_k = \left(\frac{b^2}{2}\right)^{k+1} \frac{1}{(k+1)!} \exp(-b^2/2), \quad k \geq 0. \tag{4}$$

Equation (4) can be quickly manipulated to give the following double inequality on $\hat{k}$, i.e.

$$\frac{b^2}{2} - 2 < k \leq \frac{b^2}{2} - 1$$

Since $\hat{k}$ must be a positive integer, we can write

$$\hat{k} = \left[\frac{b^2}{2} - 1\right], \quad b \geq 2$$

where the brackets denote the operation of taking the largest integer less than or equal to the argument.

To determine how much error is made in truncating the sum in eqn (2), notice that

$$\sum_{k=x}^{y} h_k f_k < \sum_{k=x}^{y} f_k \quad \forall x, y \quad 0 \leq x \leq y \leq \infty \tag{5}$$

because $0 < h_k \leq 1$ and $f_k > 0 \forall k \geq 0$. Thus, the r.h.s. of Ineq. (5) serves as an upper bound on the sum in eqn (2). We use this observation to compute an upper bound on the truncation error made by using a finite number of terms of the sum. Calling this error e_n, and if we retain only the first n terms, then we can write

$$\begin{aligned} e_n < \sum_{k=n}^{\infty} f_k &= \sum_{k=n}^{\infty} \left(\frac{b^2}{2}\right)^{k+1} \frac{1}{(k+1)!} \exp(-b^2/2) \\ &= \left(\frac{b^2}{2}\right)^{n+1} \frac{1}{(n+1)!} \exp(-b^2/2)\left[1 + \left(\frac{b^2}{2}\right)\frac{1}{n+2} + \left(\frac{b^2}{2}\right)^2 \frac{1}{n+3} \cdot \frac{1}{n+2} + \cdots\right] \\ &< \left(\frac{b^2}{2}\right)^{n+1} \frac{1}{(n+1)!} \exp(-b^2/2) \cdot \left[1 + \frac{b^2}{2n} + \left(\frac{b^2}{2n}\right)^2 + \cdots\right] \end{aligned}$$

If $(b^2/2n) < 1$, then we can sum the geometric series in the brackets to obtain

$$e_n < \left(\frac{b^2}{2}\right)^{n+1} \frac{1}{(n+1)!} \cdot \frac{\exp(-b^2/2)}{1-(b^2/2n)}, \quad n > \frac{b^2}{2} \tag{6}$$

In fact, if we always retain the first $\hat{k}+1$ terms of the sum in eqn (2), i.e. all terms with index such that $0 \leq k \leq \hat{k}$, then the constraint on n in Ineq. (6) will be automatically satisfied. Since we have already shown that these terms form an *increasing* sequence, a practical computer sub-routine to calculate $Q(a, b)$ would no doubt have this feature in any case.

If we denote the maximum allowable error in computing $Q(a, b)$ by E, then requiring that

$$\left(\frac{b^2}{2}\right)^{n+1} \frac{1}{(n+1)!} \cdot \frac{1}{1-(b^2/2n)} < E \exp(b^2/2) \tag{7}$$

insures that $e_n < E$. A stronger condition is

$$\left(\frac{b^2}{2}\right)^{n+1} \frac{1}{(n+1)!} < E \exp(b^2/2) \tag{8}$$

From the arithmetic mean-geometric mean inequality [9], we can write

$$(n+1)! < \left(\frac{n+2}{2}\right)^{n+1}$$

Using this result in Ineq. (8), we obtain the even stronger condition, which is our main result,

$$\left(\frac{b^2}{n+2}\right)^{n+1} < E \exp(b^2/2) \tag{9}$$

The value of n that first makes Ineq. (9) true is a *lower* bound on the n that first makes Ineq. (7) true. That is, if we solve Ineq. (9) for the smallest n that makes the inequality true, then *at least*

that value of n must be used to insure that the *upper* bound on e_n, given by Ineq. (6), is less than E. In fact, however, e_n itself may be less than E for a smaller n, a situation that occurs for "small" values of b. An example of this is given in the next section for the case $b = 0.1$.

SOME NUMERICAL RESULTS

To experimentally study the tightness of the bound given by Ineq. (9), McGee's algorithm was programmed and run on a DEC System-10, using double precision FORTRAN (18 significant digits). For each of several choices of arguments the program was allowed to run, printing out each successive iteration, until the computed value of $Q(a, b)$ stabilized. This value was taken as the actual value of $Q(a, b)$. From this list of numbers, given an E, one can then find at which iteration the computed value was first within E of the stabilized value. In the following table, Q_n denotes the computed value when n terms are retained in the sum of eqn (2) and represents the first occurrence for which $Q_n - Q \leq E$. The rightmost column is the value of n given by Ineq. (9).

Table 1

a	b	Q	Q_n	n	Ineq. (9)
0.1	5	0.396264015089(−5)	0.409786952243(−5)	34	26[1]
2	5	0.222082973713(−2)	0.222096496650(−2)	34	26[1]
6	5	0.862514836230(0)	0.862514971432(0)	34	26[1]
9	5	0.999976870481(0)	0.999976999129(0)	32	26[1]
5.1	0.1	0.999999988414(0)	0.999999988419(0)	2	4[2]
8	3	0.999999829129(0)	0.999999829215(0)	21	20[2]
8	8	0.524983026691(0)	0.524983026850(0)	73	52[2]
1	10	0.179977560632(−16)	0.179977560632(−16)	125	94[3]

(1) $E = 2(-7)$, (2) $E = 2(-10)$, (3) $E = 2(-20)$.

CONCLUSION

The examples given in the table can hardly be taken as exhaustive, but they do show that Ineq. (9) gives a reasonably good estimate of the number of terms required for a specified accuracy. The value of having such estimates is that they allow one to roughly calculate the CPU time required to support many calls to a subroutine implementing McGee's algorithm. A common practice in coding such iterative algorithms is to let the subroutine loop until successive iterations differ by less than some value, e.g. $E/10$. Without the guidance of an error analysis, such a technique is an ***unknown*** in its run time as a function of E.

Finally, a comment about McGee's algorithm when implemented on a "low" accuracy system. When written in RTB (Real-Time BASIC[10]), on a Varian 620/L-100 using seven significant digits, $Q(6, 5)$, for example, stabilized at 33 terms with a computed value of 0.862512(0). As can be seen by comparing this result with the entry in the table, this result, while possibly not too bad in accuracy, is ***theoretically*** wrong as it ***underestimates*** the correct value, a phenomena not inherent in the algorithm.

REFERENCES

1. G. Arfken, *Mathematical Methods for Physicists*. Academic Press, New York (1966). In particular, see relations 11.146 and 11.147 on p. 405.
2. C. W. Helstrom, *Statistical Theory of Signal Detection*, 2nd Edn. Pergamon Press, New York (1968).
3. S. O. Rice, The mathematical analysis of random noise. *Bell. Syst. Tech. J.* **23**, 282–332 (1944); **24**, 46–156 (1945).
4. J. I. Marcum, A statistical theory of target detection by pulsed radar. *IRE Trans. Inform. Theory* **6**, 58–268 (1960).
5. J. I. Marcum, *Table of Q-functions*. U.S. Air Force Project Rand Research Memo RM-339 (1950).
6. L. Maisel, Performance of sidelobe blanking systems. *IEEE Trans. Aerospace Sys.* **4**, 174–180 (1968). In particular, see Maisel's eqn (8).
7. L. E. Brennan and I. S. Reed, A recursive method of computing the Q-function. *IEEE Trans. Inform. Theory* **11**, 312–313 (1965).
8. W. F. McGee, Another recursive method of computing the Q-function. *IEEE Trans. Inform. Theory* **16**, 500–501 (1970).
9. E. Beckenbach and R. Bellman, *An Introduction to Inequalities*. L. W. Singer Company (1961). In particular, see inequality (4.19).
10. C. L. Wilkins, *Digital Electronics and Laboratory Computer Experiments*. Plenum Press, New York (1975).